DEVELOPING AN ARCHITECTURE FOR A 3 TIER WEB APPLICATION ON AMAZON WEB SERVICES

BY

GERTRUDE CHICHI

APRIL 25, 2025

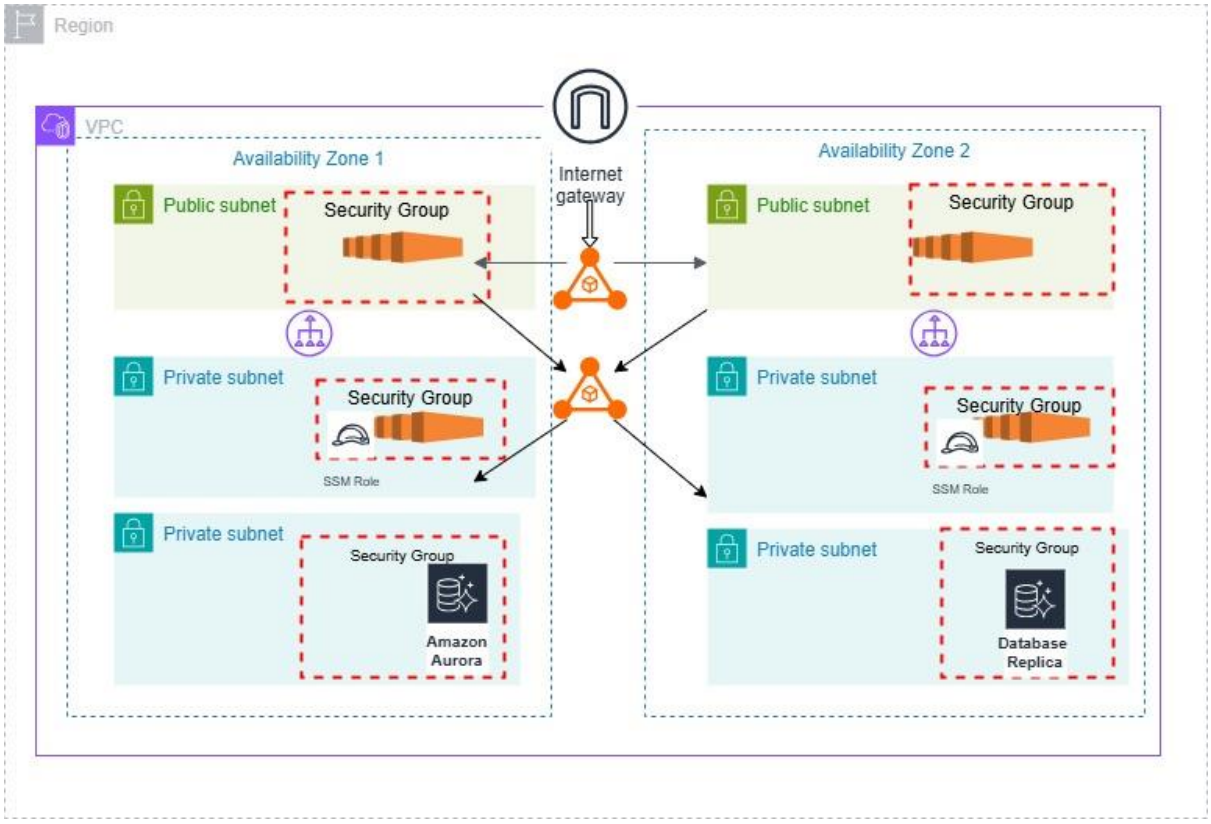
DECLARATION

I affirm that this document represents the results of my findings in exploring AWS resources by embarking on a hands-on project in architecting scalable and highly available infrastructure on AWS. This work is authentic and was developed after thoroughly referencing AWS documentation and reviewing existing solutions for building cloud infrastructure during my training session with Azubi Africa.

INTRODUCTION

As a solutions architect there is a need to build a robust architecture by considering factors such as elasticity, availability and security. This hands-on lab work focused on building a robust AWS infrastructure that can host applications from clients and make users satisfied with the application hosted in the cloud.

AWS 3-Tier Application Design: Web, App, and Database Layer

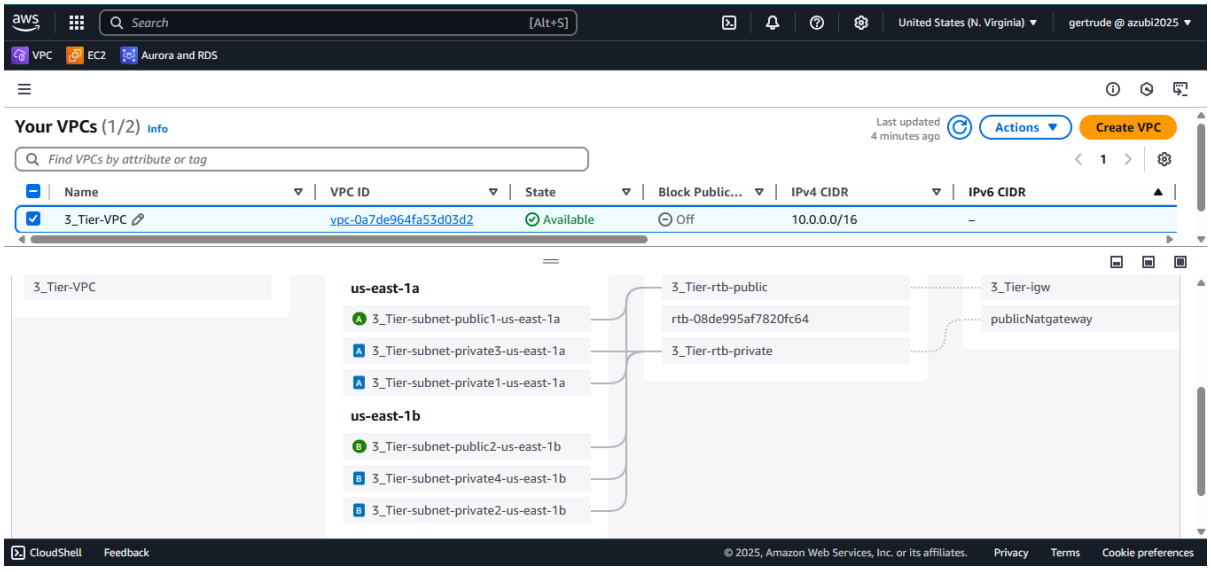


AWS THREE (3) TIER APPLICATION- STEP BY STEP CONSOLE CONFIGURATION

Virtual Private Cloud (VPC)

The VPC ((3_Tier-VPC) was made up of two availability zones in the us-east-1 region. The subnets were built in the us-east-1a and us-east-1b availability zones. Each availability zone had a public subnet and two private subnets. In all, the VPC comprised of six subnets in the two availability zones.

The infrastructure utilized two route tables. One of the route table (3_Tier-rtb-public) connected the public subnets to the internet gateway. For the private subnets which were four in number, they were all associated with the private route table (3_Tier-rtb-private). The NAT gateway was also associated with the private route table to ensure that the private subnets could send outbound traffic to the internet. I auto enabled public IPV4 address for the two public subnets so that any resource provisioned in that subnet will automatically have public ipv4 address.

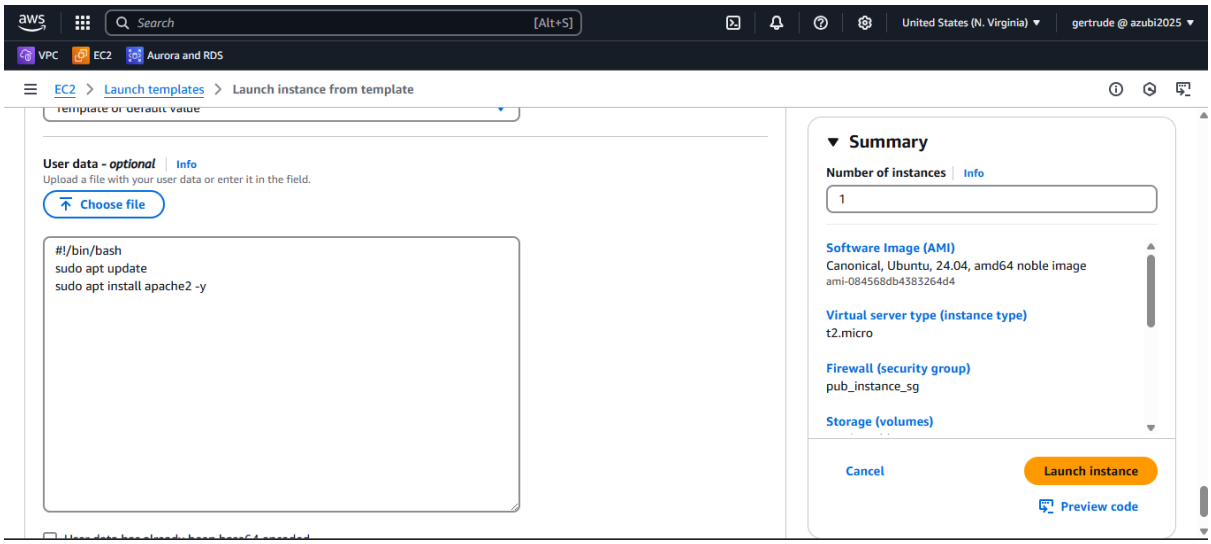


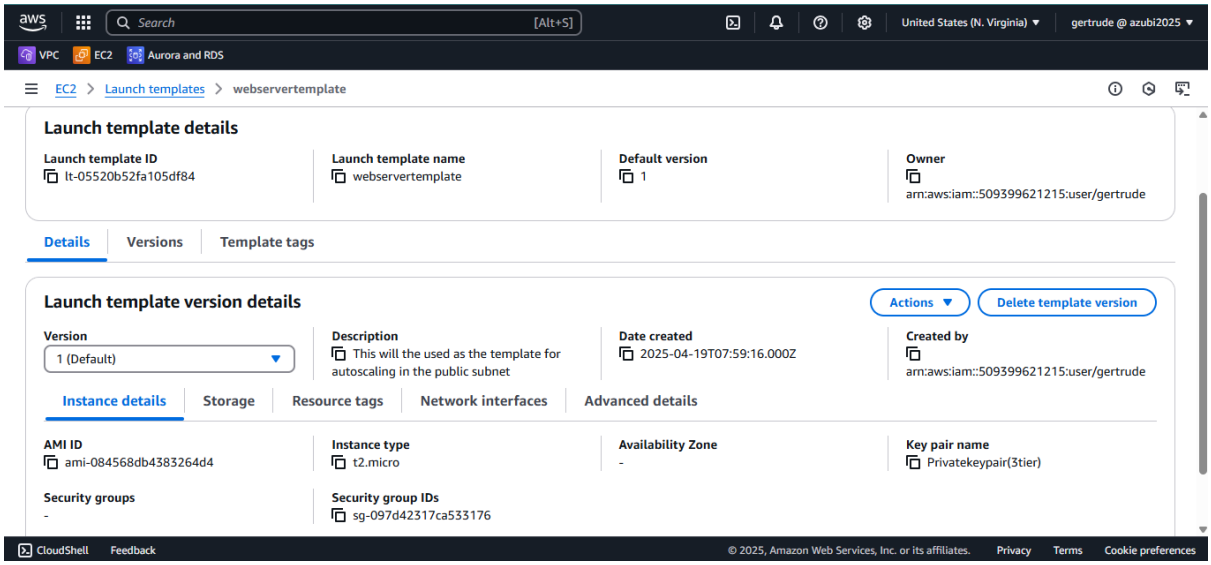
WEB-TIER

The web tier acts as the public-facing interface, serving content to users.

Launch Template: The webserver template was configured and served as a reference for the provisioning of EC2 instances in the two availability zones of the public subnets named 3_Tier-subnet-public1-us-east-1a and 3_Tier-subnet-public2-us-east-1b.

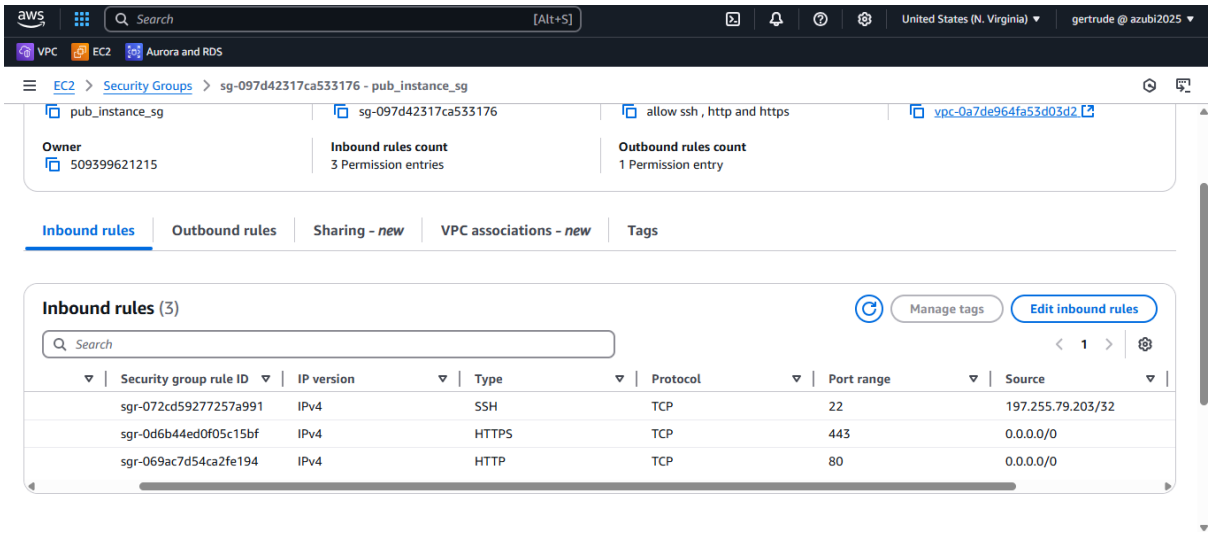
The instance type that was used was t2.micro and it was free tier eligible .In the user data of the template , a bash script for updating the sever as well as installing apache sever was included and hence, the EC2 instance provisioned required no updates and installation of the apache2 server since the template provided a configuration that enabled the update and installation of apache websever on the EC2 instance.





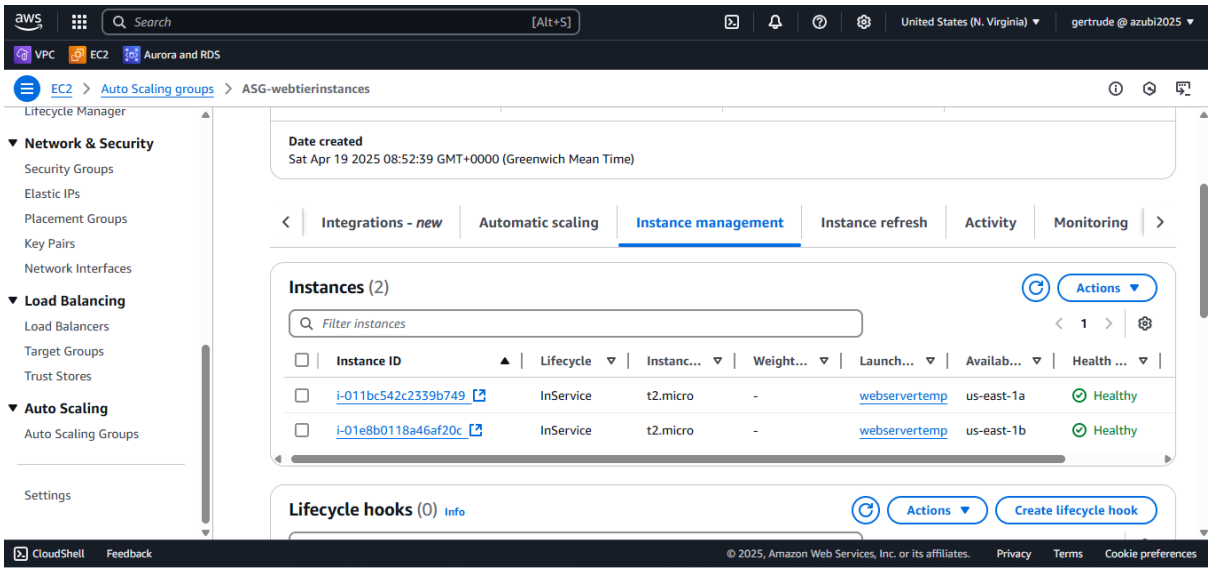
Through the websever configuration, I also honed my skill in hosting a static website leveraging AWS EC2 instance.

Security group: The security group for the ec2 instances in the public subnet was configured to allow HTTPS and HTTP from anywhere and SSH from my IP only which allowed only my IP address to access the instance and run updates and other installation.



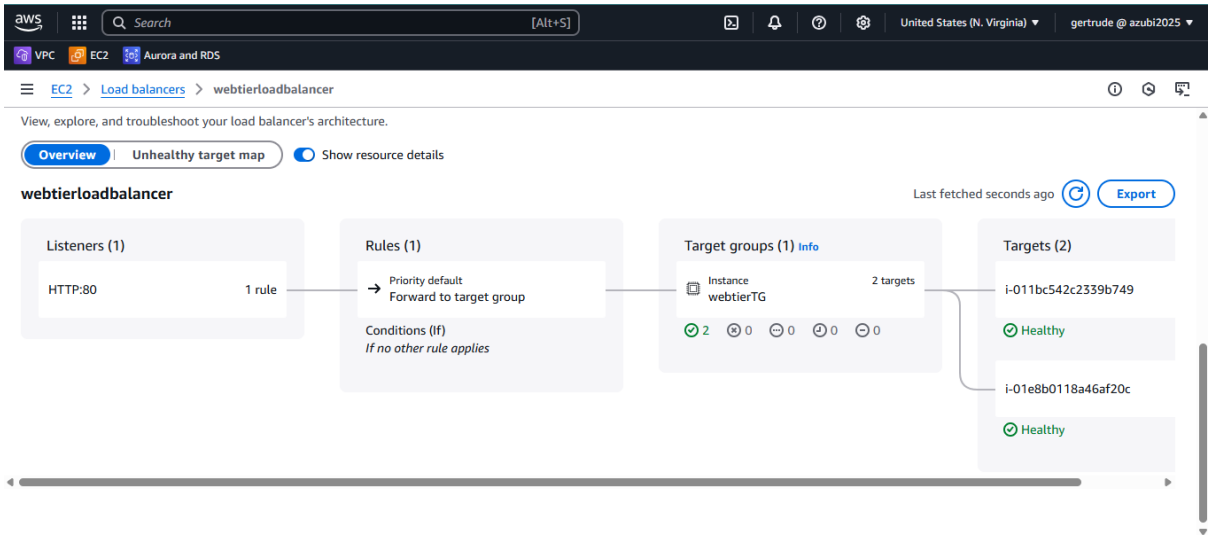
Autosacling group: For the autoscaling group. The webtemplate served as a reference which allowed the autoscaling group to launch instances.The desired capacity of the instances to be

launched by the autoscaling group was 2, the minimum desired capacity was,1 and the maximum desired capacity was 2 as well.

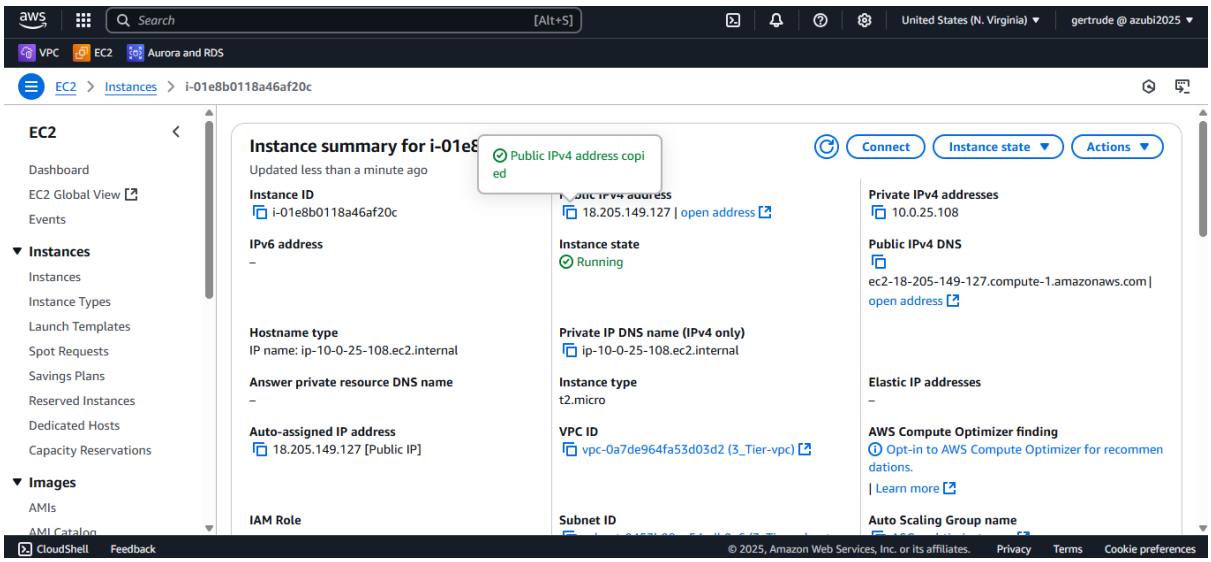


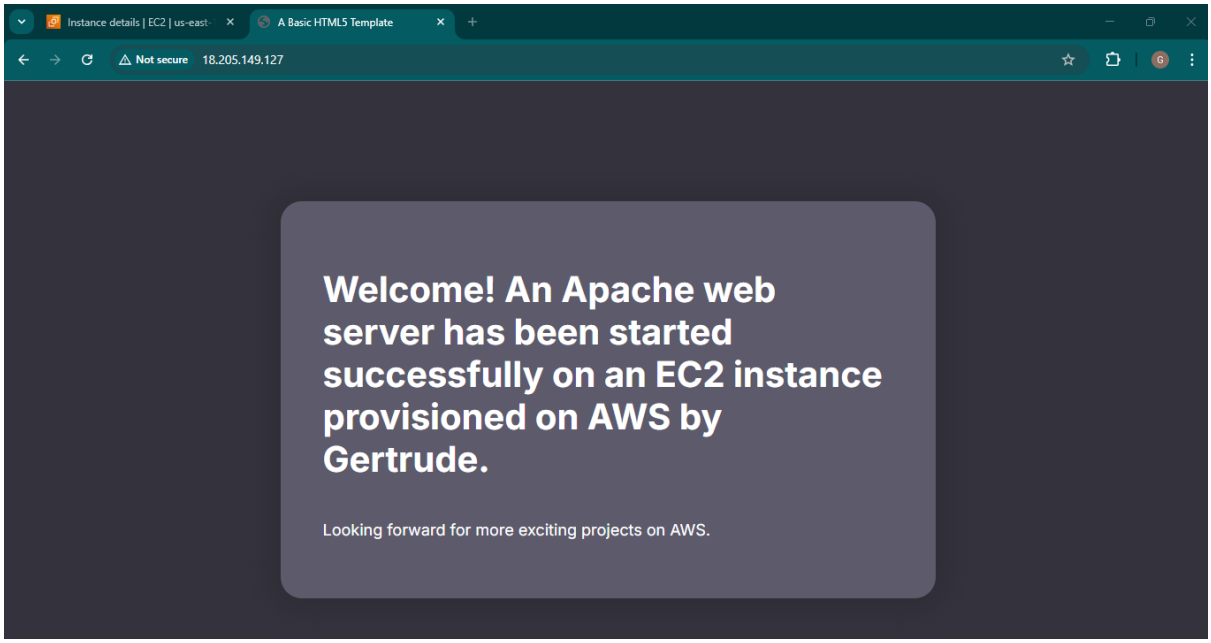
Load balancer: Ec2 instances were provsioned in the two availability zones of the public subnets when the load balancer directed traffic from the internet to the target group . The routetable ensured that traffic from the internet was routed to the public subnets.

The purpose of the loadbalancer was to direct traffic to the availble EC2 instances which served as the target group in the public subnet.The HTTP and HTTPS traffic from the internet was directed to the load balancer because the security group of the load balancer allowed HTTP and HTTPS traffic.To allow evenly didtribution of traffic and to prevent overwhelming of a single webserer with request from users, the application loadbalancer was needed.The EC2 instances remained healthy after the distribution of the HTTP and HTTPS traffic to these insatnces.

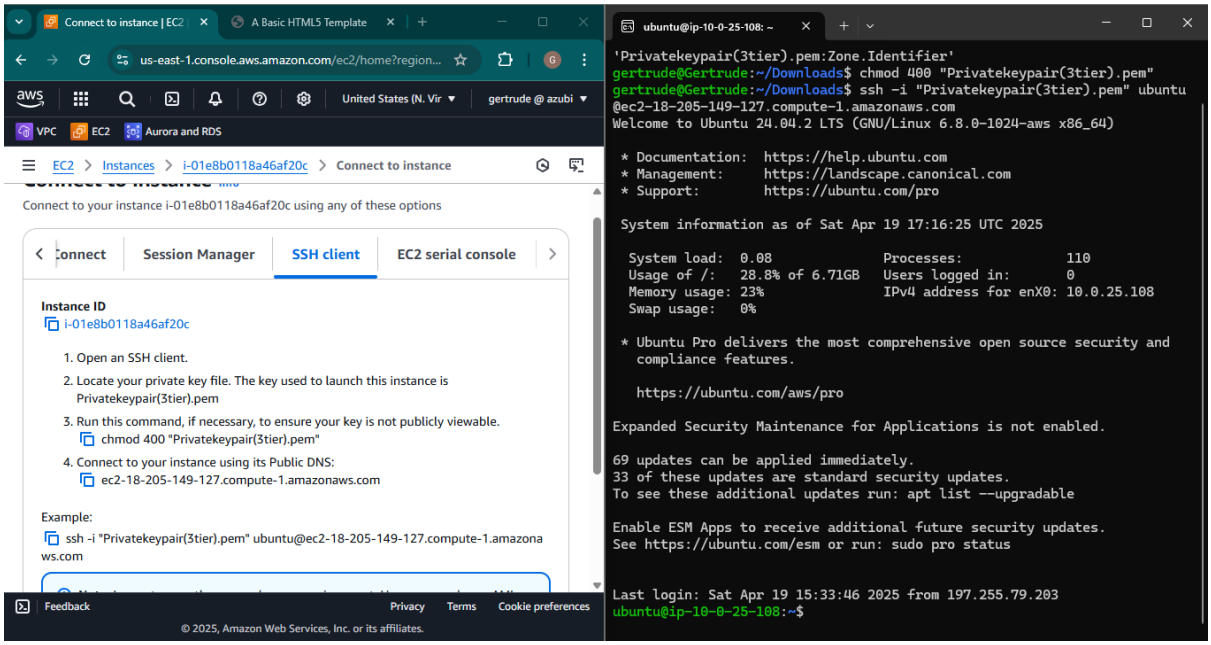


After hosting a webserver in one of the provisoned ec2 instance from the autoscaling group, I copied the public IPV4 address of the EC2 instances and pasted it in the web browser where I confimed it was accessible by the public .





To confim that the aspect of the security group of the web server that allowed inbound traffic of ssh from my IP was applied to the autoscaling group I connected to the EC2 instance using my IP on a ubuntu terminal and it was successful.

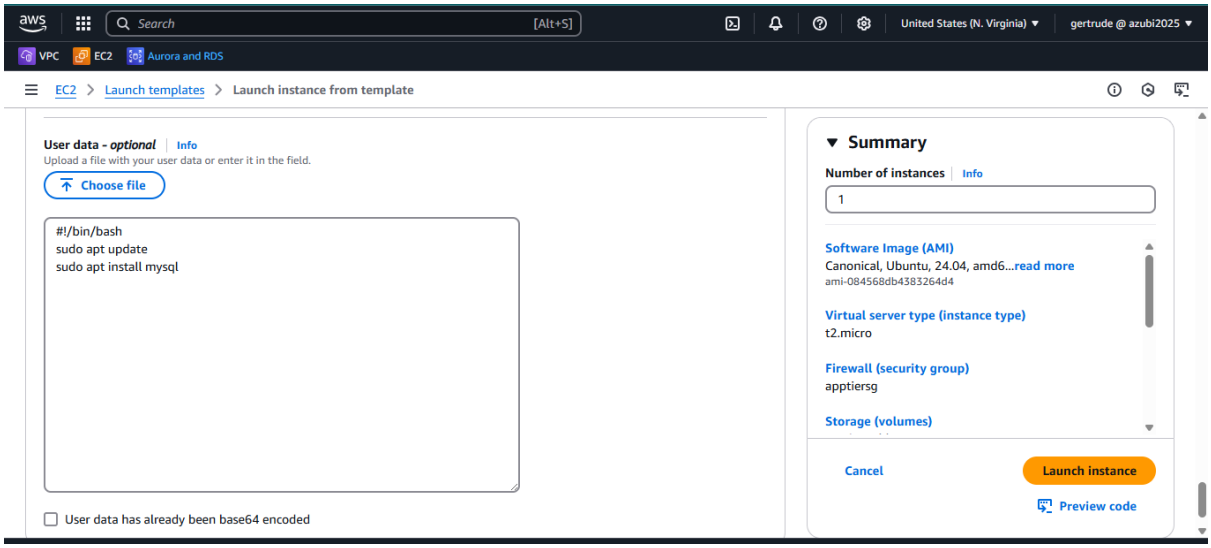


To conclude, for the web tier, I was able to successfully provision EC2 instances using the application load balancer, autoscaling group with the necessary security groups attached to the load balancer and the launch template which allowed HTTP, HTTPS from the internet and SSH from my IP.

APP-TIER

The app servers were hosted in the private subnet and served as the client to the database instance.

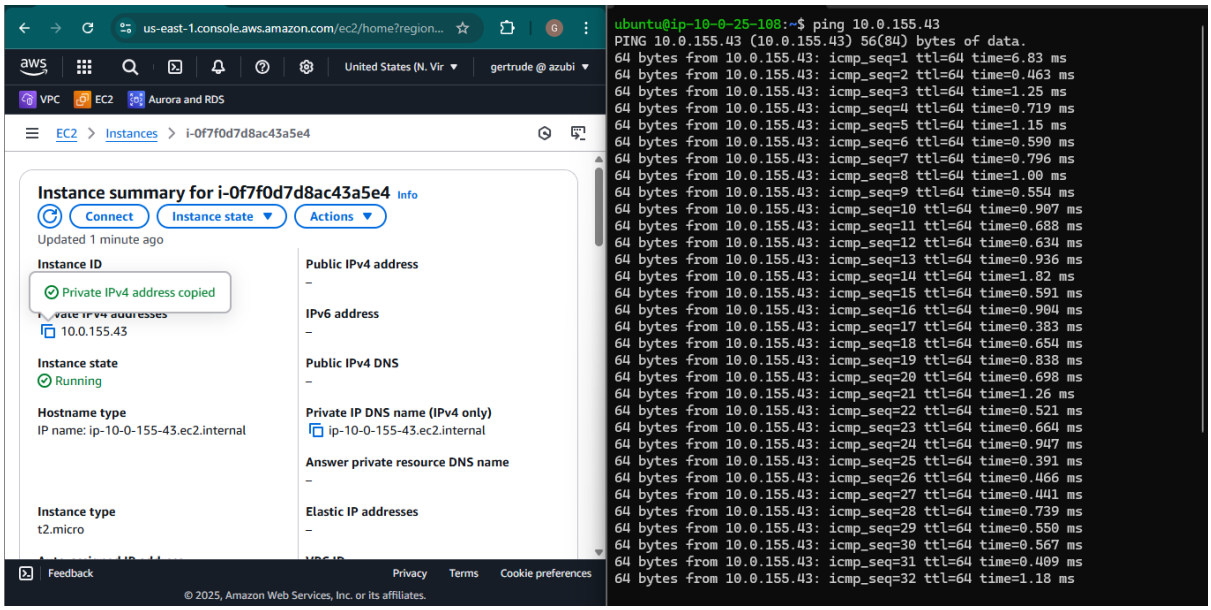
Launch template: I also configured a launch template for the provisioning of EC2 instances in the app tier. The template used for launching of the ec2 instances in the app tier was similar to the template that was used in the web tier taking into consideration the instance type that was used. Attached to the template was a user data for the installation of mysql client upon provisioning of the EC2 instance.



The major difference between the webserver instances and the app server instances lied in the security groups that was attached to each of them. The load balancer for the app server was also an internal load balancer directing traffic within the VPC.

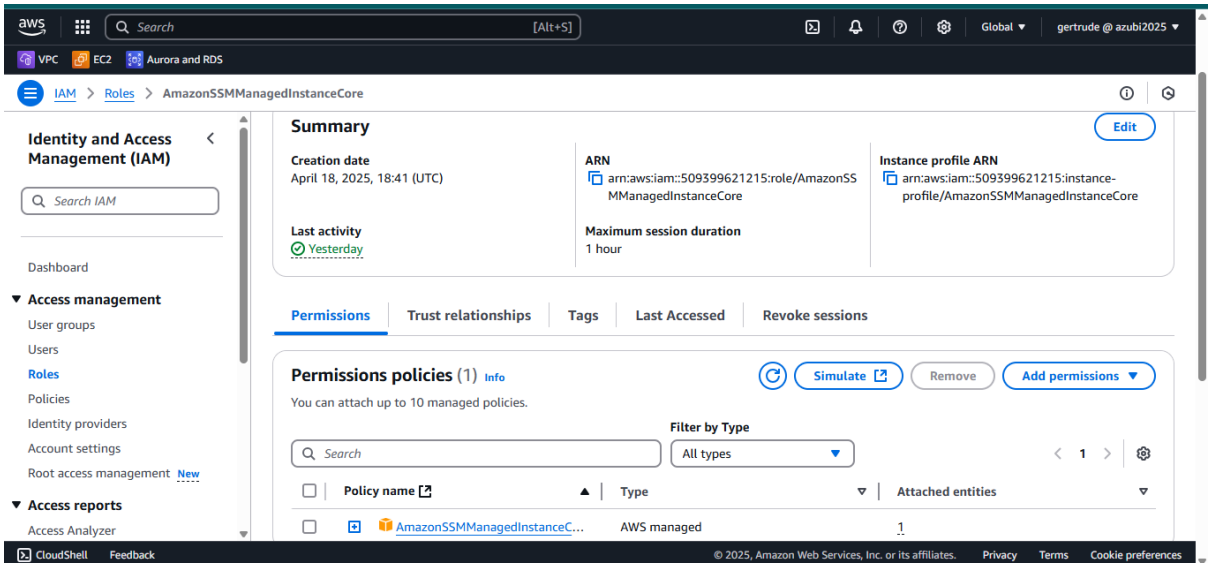
To ensure that the instances were not publicly available. The app tier instances were provisioned in the private subnet allowing only inbound traffic from the instances in the public subnet (security group attached to the web tier instance). An internal load balancer which was not internet facing was created. The purpose of the internal load balancer was to distribute traffic within the VPC. The internal load balancer listened to port 80 of the app server target group.

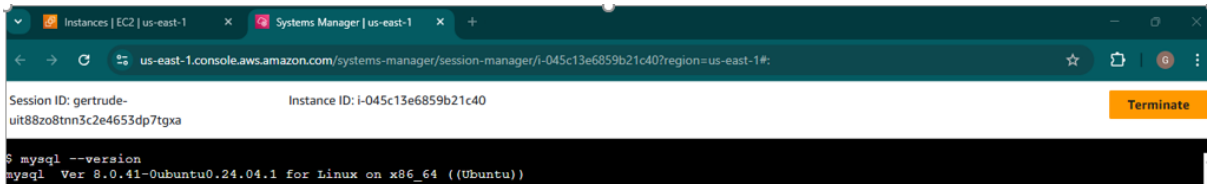
The instances in the app tier were launched using the autoscaling principle and an internal load balancer. The web server security group was configured send ICMP echo requests to the app servers and the app servers responded to the ICMP echo requests. Due to this, I could confirm that there was a secure connection between the instances in the webserver and the instance in the app tier (all resided in the VPC) .



However, when the app server instances were in service, I was unable to SSH into the app server to run administrative tasks and confirm if mysql client has been installed in the app server even though I could verify that there was a secure connection between the app server instances and the instances in the web tier confirming the availability of app instances.

To rectify the issue, I created a role with the SSM Manged instance core permission and attached it to the app severs which allowed connection to the instances in the app tier and confirm the installation of the mysql client on the app server


Search [Alt+S]
Global
gertrude @ azubi2025
VPC
EC2
Aurora and RDS
IAM > Roles > AmazonSSMManagedInstanceCore
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management New
Access reports
Access Analyzer
Summary
Edit
Creation date
April 18, 2025, 18:41 (UTC)
ARN
arn:aws:iam::509399621215:role/AmazonSSMManagedInstanceCore
Instance profile ARN
arn:aws:iam::509399621215:instance-profile/AmazonSSMManagedInstanceCore
Last activity
Yesterday
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (1) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
Policy name
Type
Attached entities
AmazonSSMManagedInstanceC...
AWS managed
1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Instances | EC2 | us-east-1
Systems Manager | us-east-1
us-east-1.console.aws.amazon.com/systems-manager/session-manager/i-045c13e6859b21c40?region=us-east-1#:
Session ID: gertrude-uit88zo8tnn3c2e4653dp7tgxa
Instance ID: i-045c13e6859b21c40
Terminate
$ mysql --version
mysql Ver 8.0.41-0ubuntu0.24.04.1 for Linux on x86_64 ((Ubuntu))

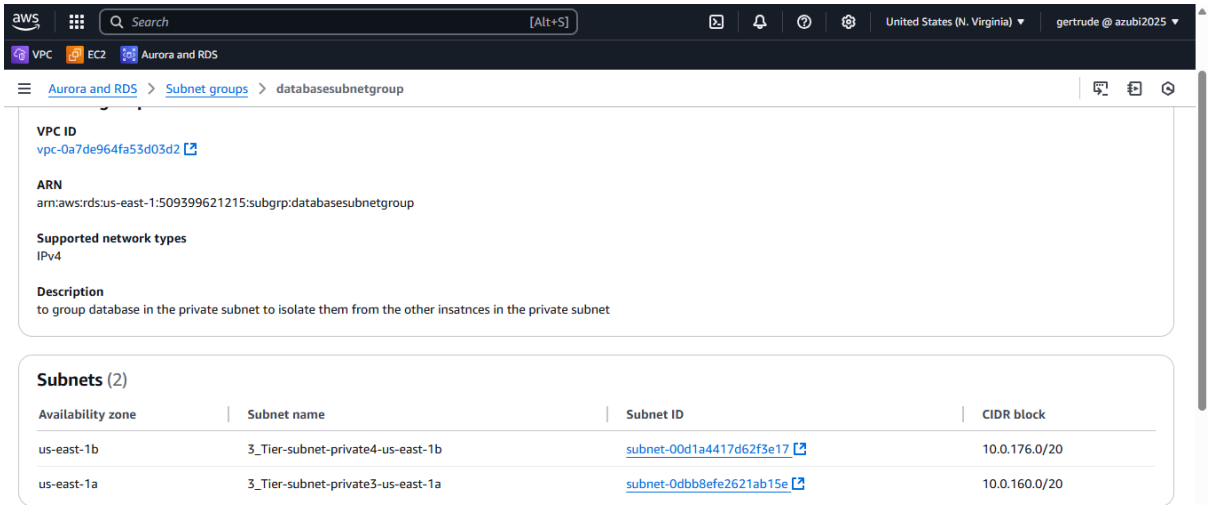
DATABASE-TIER

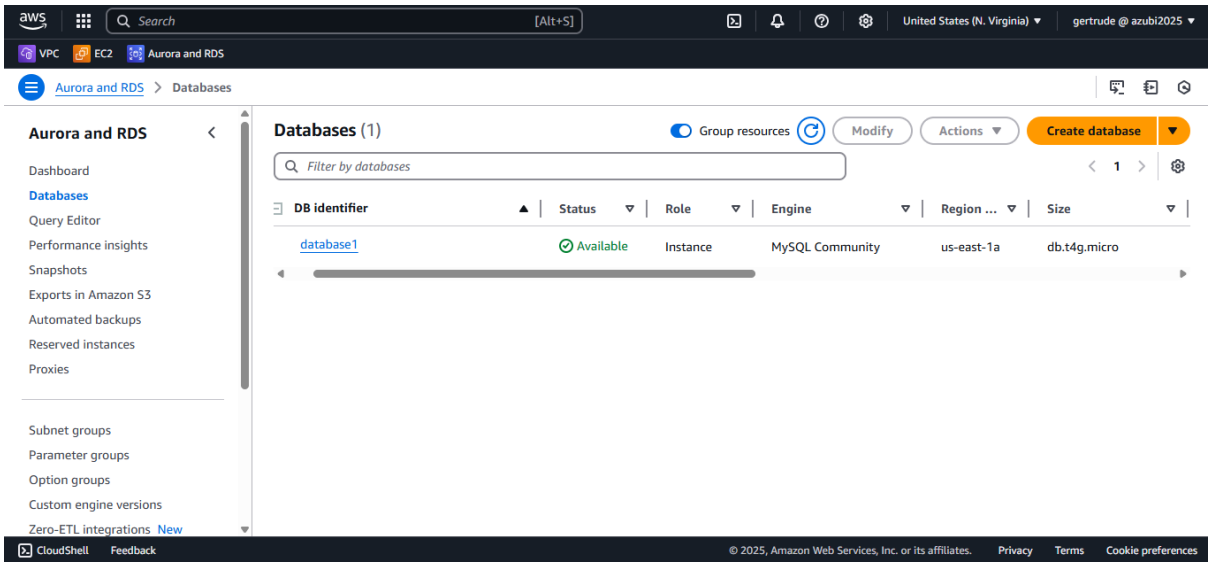
The database served as the server to the app servers

Security group: The security group of the app server allowed inbound MySQL /aurora traffic from the database security group and database security group also was configured to outbound MySQL aurora traffic to the data base client which was installed on the app server.

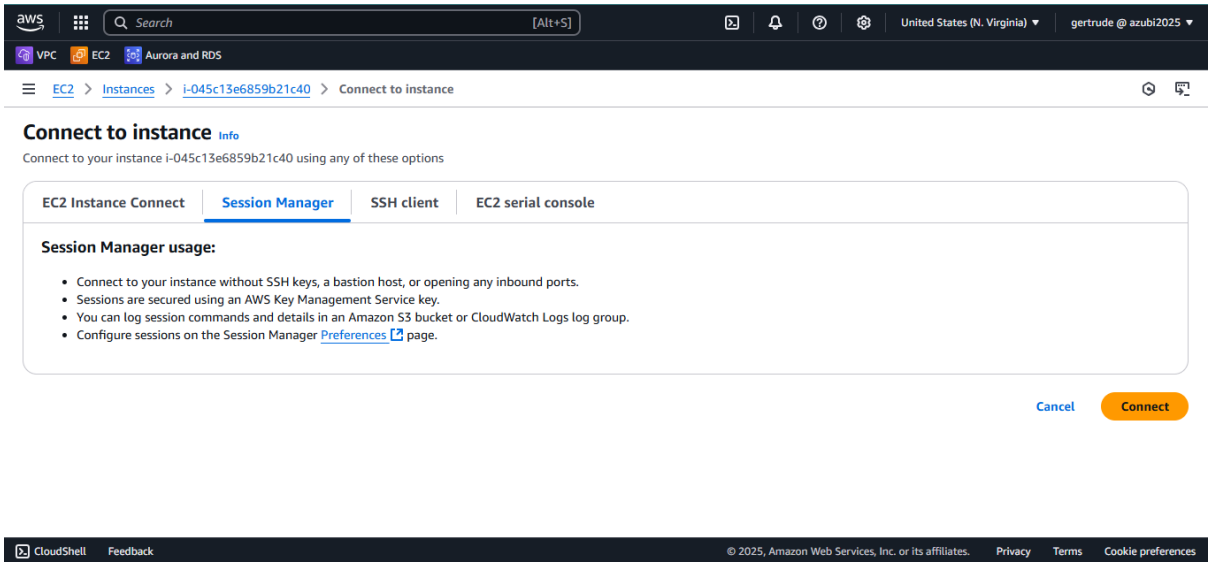
This client-server model between the app servers and the database instance facilitated the sending of query results to the app server from the database instance and retrieval of database query results from the database instances to the app server.

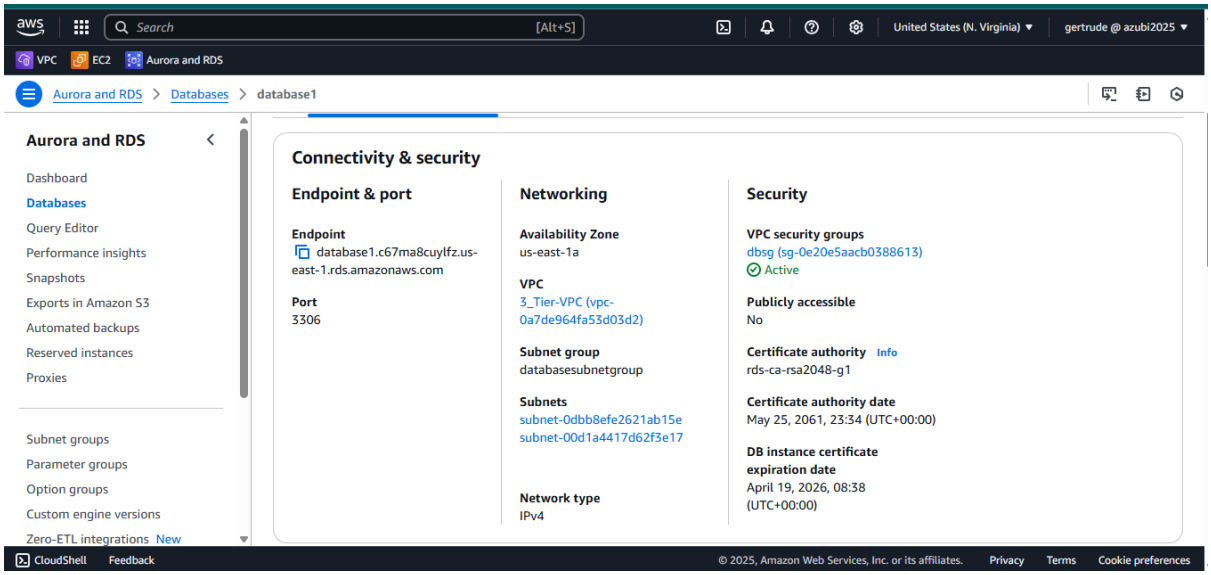
Database Subnet Group: A database subnet group was created to isolate the database instances within the private subnet. For the creation of the database, MySQL engine was selected because it was compactible with the MySQL client that was installed on the app server.

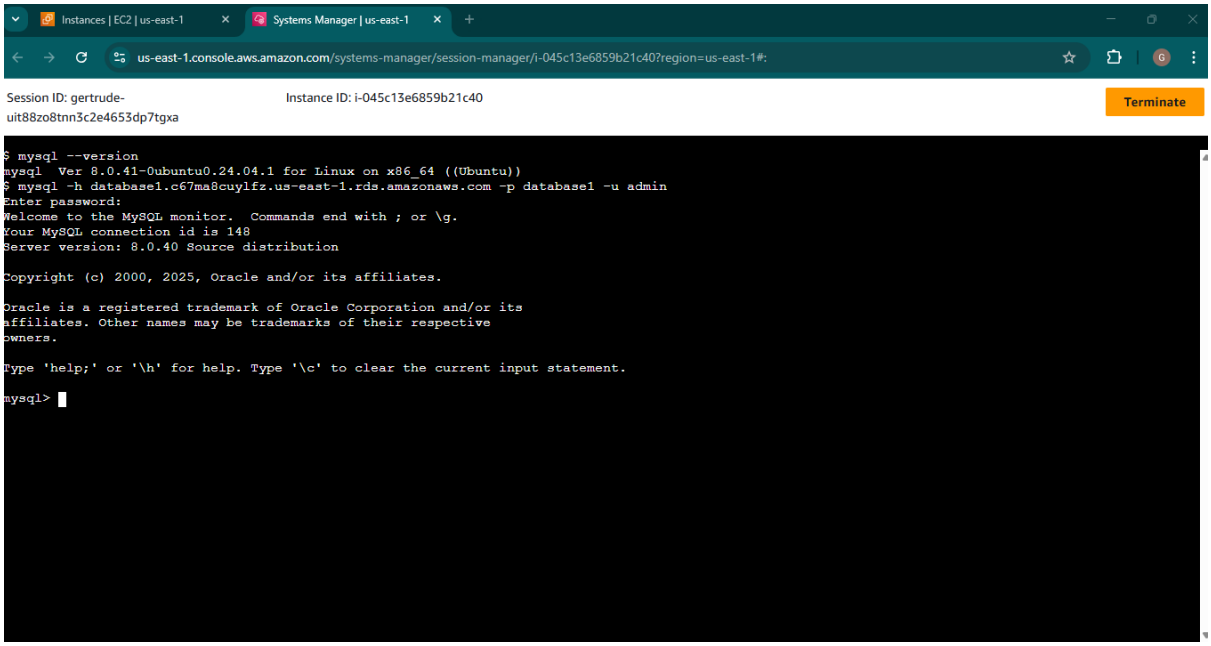




I successfully connected to the app server using the Session manager. The endpoint of the database was used to connect to the MySQL client successfully.


Search
[Alt+S]
United States (N. Virginia)
gertrude @ azubi2025
VPC
EC2
Aurora and RDS
EC2 > Instances > i-045c13e6859b21c40 > Connect to instance
Connect to instance Info
Connect to your instance i-045c13e6859b21c40 using any of these options
EC2 Instance Connect
Session Manager
SSH client
EC2 serial console
Session Manager usage:
Connect to your instance without SSH keys, a bastion host, or opening any inbound ports.
Sessions are secured using an AWS Key Management Service key.
You can log session commands and details in an Amazon S3 bucket or CloudWatch Logs log group.
Configure sessions on the Session Manager Preferences page.
Cancel
Connect
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search
[Alt+S]
United States (N. Virginia)
gertrude @ azubi2025
VPC
EC2
Aurora and RDS
Aurora and RDS > Databases > database1
Aurora and RDS
Dashboard
Databases
Query Editor
Performance insights
Snapshots
Exports in Amazon S3
Automated backups
Reserved instances
Proxies
Subnet groups
Parameter groups
Option groups
Custom engine versions
Zero-ETL integrations New
Connectivity & security
Endpoint & port
Endpoint
database1.c67ma8cuylfz.us-east-1.rds.amazonaws.com
Port
3306
Networking
Availability Zone
us-east-1a
VPC
3_Tier-VPC (vpc-0a7de964fa53d03d2)
Subnet group
databasesubnetgroup
Subnets
subnet-0dbb8efe2621ab15e
subnet-00d1a4417d62f3e17
Network type
IPv4
Security
VPC security groups
dbsg (sg-0e20e5aacb0388613)
Active
Publicly accessible
No
Certificate authority Info
rds-ca-rsa2048-g1
Certificate authority date
May 25, 2061, 23:34 (UTC+00:00)
DB instance certificate expiration date
April 19, 2026, 08:38 (UTC+00:00)
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



CHALLENGES

Implementing a secure 3-tier architecture posed significant challenges, including configuring security groups to balance access and security, ensuring seamless communication between tiers, and restricting administrative access to specific IP addresses without disrupting functionality